2004

The Evolution, current status, and future direction of XML

Christina Ann Oak

The Evolution, Current Status, and Future Direction of XML

By,

Christina Ann Oak

Thesis submitted in partial fulfillment of the requirements for the degree of Master of Science in Information Technology

Rochester Institute of Technology

B. Thomas Golisano College
of
Computing and Information Sciences

June 21, 2004

Rochester Institute of Technology

B. Thomas Golisano College
of
Computing and Information Sciences

Master of Science in Information Technology

Thesis Approval Form

Student Name: Christina Ann Oak

Project Title: The Evolution, Current Status, and Future Direction of XML

Thesis Committee

Name	Signature	Date
William J. Stratton	William J. Stratton	6/21/04
Chair		
Name Illegible	Name Illegible	6/21/04
Committee Member		
Tona Henderson	Tona Henderson	6/21/04
Committee Member		

Thesis Reproduction Permission Form

Rochester Institute of Technology

B. Thomas Golisano College of Computing and Information Sciences

Master of Science in Information Technology

The Evolution, Current Status, and Future Direction of XML

I, Christina Ann Oak, hereby grant permission to the Wallace Library of the Rochester Institute of Technology to reproduce my thesis in whole or in part. Any reproduction must not be for commercial use or profit.

Date: 12/21/2024 Signature of Author: Christina Oak

ABSTRACT

The Extensible Markup Language (XML) is now established as a multifaceted open-ended markup language and continues to increase in popularity. The major players that have shaped its development include the United States government, several key corporate entities, and the World Wide Web Consortium (W3C). This paper will examine these influences on XML and will address the emergence, the current status, and the future direction of this language. In addition, it will review best practices and research that have contributed to the continued development and advancement of XML.

TABLE OF CONTENTS

CHAPTER I:

INTRODUCTION

The Extensible Markup Language (XML) allows developers to create systems that capture, manipulate, and exchange a wide variety of documents and data. As many authors have noted (e.g. Kim; Phillips; and Tittel, Pitts, and Boumphrey) there are numerous experts who believe that XML represents a kind of 'lingua franca' (or language) that can signify numerous types of information. XML does this in a way that makes that information more accessible to human readers, as well as all kinds of computer applications and services. XML has become known as one of the most significant developments in the history of computing.

The use of markup conventions dates back to the Ancient Egyptians. Over time, markup conventions have grown and improved, with the most recent introduction being XML. XML has proven successful in fulfilling the original design goals set forth by the World Wide Web Consortium (W3C) XML Working Group. These design principles have contributed immensely to the success and fast growing use of XML and its popularity is likely to continue growing over the next several years.

As a result of its success, many major corporations with significant resources have invested in XML. Among them are some of the heavy hitters in the industry, including Microsoft, IBM, SAP, Netscape, Oracle, Sun Microsystems, the U.S. government, and the U.S. military. These corporations and many others have discovered that XML isn't limited to its use on the web; it can be used at every level of an enterprise.

The World Wide Web is undergoing a massive transformation as a result of many factors, including the fast growing use of XML and its related technologies. At this critical time in the development of XML, standards are being proposed and propagated faster than any one can follow. There are a number of proposals from W3C members, alternatives from people with no affiliation at all, user group initiatives, and attempts to reconcile them all. New XML-based languages are also being developed.

This paper will explore an understanding of these new developments. It will focus on the evolution of XML, which up to now, has not typically been the focus of much research. This paper will also address the current status and future direction of XML, including a review of best practices and research that have contributed to the continued development and advancement of XML.

CHAPTER II:

THE EMERGENCE OF XML

The History of Markup Conventions

Ancient Egyptian Markup Conventions

Markup has been around almost since the beginning of written languages. The Egyptians, who used one of the most ancient writing systems in the world, hieroglyphic symbols, marked personal names with an oval cartouche to distinguish them from ordinary words. They also used color to highlight important phrases, similar to current boldface. Since the Egyptian language could be written from left to right or right to left, depending on the context, they faced the pictures of animals and people toward the start of the line as an antique bi-directional attribute and markup. The Egyptians did this because they thought it was more polite to have the figures greet the reader as he/she read along the line. They felt it would be rude if the figures faced away (Phillips).

Ancient Hebrew Markup Conventions

The ancient Hebrew language contains almost no markup at all except for decorative 'crowns' over seven letters and two final letter forms. Like many writing systems, ancient Hebrew came late to the idea of putting spaces between words. It also did not use any indication of vowels, preferring to supply them by a process of sheer reason (Phillips). However, around the 8th century, a system called 'Nikud' (points) was developed in order to aid in the pronunciation of vowel sounds in the Hebrew language. This 'markup' consists of small dots and dashes that are placed above, below, or inside the letter ("All About the Hebrew Alphabet"). Nowadays, it is still

common to find these 'points' used in scriptures, children's books, and also poetry. However, books, magazines, and newspapers that are published in Israel today are still written in a language similar to that of the ancient Hebrew language.

Ancient Roman Markup Conventions

The history of our own Roman lettering is similar to both the Egyptian and Hebrew markup conventions. It includes such elements as word spacing, rhetorical punctuation, the introduction of lowercase letters to make the difference between ordinary words and important ones more obvious, and similar innovations which are designed to make writing easier to understand and use. Nonetheless, a markup system that is many hundreds, even thousands of years old, has been used for quite some time (Phillips).

Over time, printers and typographers developed a clever system of markup that was used to identify and call for formatting changes to parts of books and other printed material. The printer's markup is actually called markup because it was added by hand, or 'marked up', on typewritten or manuscript pages. Human typesetters followed written markup instructions such as those for special marks, boldface, italic, small caps, indentations and spaces of various sorts, font changes, special treatments of titles, chapter headings, footers, headers, page numbering, and many others (Phillips).

Recent Markup Conventions

With the advent of the computer age came the development of text processing systems which encompass original document production, as well as document storage,

formatting, manipulation, publishing and other tools that parse, render, fold, spindle, and mutilate text. Recent advancements in word processing and desktop publishing systems, such as MicroPro's WordStar, Corel's WordPerfect, and Microsoft Word are designed to mask the complexity of the text formatting tasks that go on underneath the surface. However, the same markup process that human typesetters used prior to technological advancements in word publishing occurs today in Word and other text processing environments (Navarro, White, and Burman).

Early formatting languages

The transition from paper and pen-based markup to today's electronically marked-up versions has gone through several different phases. One of the earliest formatting languages, called Runoff, was coded by Jerry Saltzer for the CTSS operating system on the IBM 7094 in the early 1960s and was later ported to Multics, another early operating system. Runoff and all of its descendents use special markup tags to control the formatted appearance of text (Phillips).

Troff and T_EX, were two other early formatting languages that did an extraordinary job at formatting printed documents; however both lacked any sense of structure. Generic coding, which is a concept that uses descriptive tags rather than formatting codes, eventually solved this problem (Ray). Generic coding was the result of the efforts of numerous individuals, particularly those in the legal profession and the book and graphic arts industries, who understood that existing incompatibilities among different sets of specific coding systems (most of which were developed by the emerging word processing industry) had lead to a number of inefficiencies. The

development of the Extensible Markup Language (XML) eventually drew from some of the basics of generic coding (White, Quin, and Burman).

The first organization to explore the idea of generic coding was the Graphic Communications Association (GCA). In the late 1960s, the GenCode project began developing ways to encode different document types with generic tags and to assemble documents from multiple pieces (Ray). This began the development of present day markup languages.

A markup language is a language that is used to label, organize, and categorize data or document content. Markup is what describes the document or data structure and organization. Content, such as text, images, and data, is what markup includes and is generally what is of greatest importance to humans who read or interact with data or documents (Tittel, Pitts, and Boumphrey).

Generalized Markup Language

The first major advancement made in contemporary markup languages was the Generalized Markup Language (GML), created by Charles Goldfarb, Edward Mosher, and Raymond Lorie of IBM in the late 1960s. It was intended as a solution to the problem of dealing with thousands of legal documents that were created on disparate systems using propriety markup languages, such as WordStar, WordPerfect, and Microsoft Word. Portability across the IBM Corporation was next to impossible because of these constraints (Navarro, White, and Burman).

Upon completion of their research, they identified three primary requirements for any interoperable system. One of the first requirements of GML is that the system

involved must support a common document format. At that time, document creation included text-formatting instructions; therefore it was logical to base GML on the model of document markup. The second requirement of GML was to develop domain-specific formatting. The third and final requirement of GML was to develop a means of verifying that the markup conforms to the first two requirements (Navarro, White, and Burman). The resulting markup language allowed documents that were coded in GML to be edited, formatted, and searched by different programs as a result of its content-based tags (Ray).

Inspired by the success of GML, the American National Standards Institute (ANSI) Committee on Information Processing assembled a team, with Goldfarb as the project leader. They began developing a standard text-description language based on GML but extended with new ideas contributed by many people. The GCA GenCode committee contributed their expertise as well (Ray).

Standard Generalized Markup Language

In 1980, the first working draft of the Standard Generalized Markup Language (SGML) standard was released by the ANSI Committee on Information Processing. By 1983, the new draft standard had been issued as an industry standard, GCA 101-1983. SGML was quickly adopted by the U.S. Department of Defense, U.S. Internal Revenue Service, as well as other U.S. government agencies and their contractors (Phillips).

In the years that followed, the success of SGML began to flourish. In 1986, the International Organization for Standardization (ISO) adopted SGML as an international data storage and exchange standard (ISO 8879:1986). Following, in 1988, the U.S.

Department of Defense issued the CALS (Computer-Aided Acquisition and Logistical Support) SGML standard known as MIL-M-28001 (Phillips).

SGML is currently overseen by many of the same participants who first brought about its creation. However, it now belongs to a subgroup known as JTC1/SC34 (International Organization for Standardization/International Electro-Technical Committee, Subcommittee 34) (Navarro, White, and Burman). This group concentrates on standardization in the field of document structures, languages and related facilities for the description and processing of compound and hypermedia documents.

SGML is an extremely powerful meta-language, which means its primary function is to define other specialized markup languages, including HTML, XHTML, and XML. SGML was designed to be a flexible and all-encompassing coding scheme. It incorporates a number of facilities that are intended to make it as complete as possible; however, these extra 'bells' and 'whistles' make SGML software expensive, difficult to implement, and slow and awkward to run (Tittel, Pitts, and Boumphrey).

HyperText Markup Language

In the early 1990s, the HyperText Markup Language (HTML) was developed by Tim Berners-Lee and Anders Berglund, employees of the European Organization for Nuclear Research (CERN), to describe and deliver textual documents across the Internet. CERN is the world's largest particle physics centre. It has been involved in the SGML effort since the early 1980s, when Berglund developed a publishing system to test SGML. Berners-Lee and Berglund created an SGML document type for hypertext documents that was compact and efficient. It was easy to write software for this

particular markup language and even easier to encode documents (Ray). This marks the beginning of HTML.

HTML is a non-proprietary format based upon SGML, and can be created and processed by a wide range of tools, from simple plain text editors to sophisticated WYSIWYG (What You See Is What You Get) authoring tools ("HTML Home Page"). HTML is not a general purpose tool to label and organize text or data. It uses a predetermined set of markup elements which makes HTML a closed markup language. The closed nature of HTML explains why it has gone through so many versions beginning with the public introduction in 1993 of HTML 1.0, and up to its final definition as HTML 4.01 in 1999. Each of its new incarnations represents the incorporation of new markup and capabilities to meet document designers' ever-expanding desire for more functionality (Tittel, Pitts, and Boumphrey).

Some believe that SGML's biggest success was HTML. However, HTML does not offer anywhere near the full power of SGML. It restricts authors to a finite set of tags designed to describe web pages and describes them in a fairly presentational way at best. Unfortunately, HTML does not lend itself to use beyond the single application of a web-page design (Harold and Means).

It appears that HTML was in some ways a step backward. In order to achieve the simplicity necessary for it to be truly useful, some principles of generic coding had to be sacrificed. For example, one document type was used for all purposes. This forced people to overload tags rather than define specific-purpose tags. Also, many of the tags are purely presentational (Ray).

In order to return to the ideals of generic coding, some people tried to adapt SGML for the Web. However, this proved too difficult. A smaller language which still retained the generality of SGML was required; thus was born the Extensible Markup Language (XML) (Ray).

Extensible Markup Language

History of XML

In early 1996, it became clear that HTML, which was seen as the most successful electronic document format in history because of its widespread use, was suffering from growing pains. In other words, every significant Web community wanted it to grow in a different direction. It was hard to perceive how to accomplish this without losing the essential simplicity and integration of display, hypertext, and GUI that has driven the success of HTML ("XML Fact Sheet").

Thus, the W3C began the process of designing an extensible markup language that combined the flexibility of SGML and the widespread acceptance of HTML (Birbeck, et al). In May of 1996, the W3C launched the project that has now become the W3C XML Activity, which eventually led to the W3C Recommendation ("XML Fact Sheet"). The working group that designed and refined XML consisted of researchers and experts from many parts of the computer industry, including Internet and intranet technology gurus, publishing wizards, and markup language mavens (Tittel, Pitts, and Boumphrey). The XML Activity was led by Dan Connolly, in collaboration with Jon Bosak. The working groups that the XML Activity is composed of are described individually in Appendix A.

The World Wide Web Consortium released XML 1.0 as a Recommendation in 1998. The initial adoption of XML by the Internet community was quite slow because of a bad misconception. XML was seen by many as a successor to HTML. However, when it became clear that this was not the case, and that XML is a separate technology in itself, things started to progress. The adoption of XML by industry heavyweights such as IBM, Microsoft, Sun, SAP, and Software AG also provided increased momentum to XML (Daum and Merten).

What is XML?

Strictly speaking, XML is not a markup language; it is considered by many to be a markup language toolkit. XML is a set of rules for defining semantic tags that break a document into parts and identify the different parts of the document (Harold, "XML Bible"). It provides the user with a foundation of syntactic constraints on which to build his/her own language. One constraint is that users follow the syntax rules of XML to create 'well-formed' XML documents. Another constraint is that the XML document be valid, meaning a document instance matches a document model (a document model is the blueprint for an instance of a markup language). A third constraint is that a user provides a DTD or schema, both of which describe elements and other markup objects within an XML document (Ray).

There are two important characteristics of XML that users should understand. First, XML is a meta-markup language, meaning the user makes up tags as they go along. Second, XML markup describes a document's structure and meaning. It does not describe the formatting of elements on a page. Formatting can be added to an XML

document by utilizing a style sheet, such as Cascading Style Sheets (CSS) or the Extensible Stylesheet Language (XSL). However, the document itself only contains tags that describe what is in the document, not what the document should look like (Harold, "XML Bible").

XML is an open standard, which means that it is not owned by a single company or individual. It is maintained and designed based on input from the community to fit real needs, not to satisfy a marketing agenda. There are also no licensing fees, nondisclosure agreements, partnerships, or intellectual property disputes; and it's free, public, and completely transparent (Ray).

The W3C created XML as an open-ended markup language that could easily accommodate future expansions and additions. The open-ended nature of XML makes it useful for many different things, such as data exchange or enhanced post-processing functions. The flexibility of XML has made it a necessity for exchanging data in a multitude of forms (Tittel, Pitts, and Boumphrey).

Design Goals of XML

The XML 1.0 specification is unusual in that it begins immediately with a statement of its design goals. The abstract from the XML 1.0 specification states the following:

> "The Extensible Markup Language (XML) is a subset of SGML that is completely described in this document. Its goal is to enable generic SGML to be served, received, and processed on the Web in the way that is now possible with HTML.

XML has been designed for ease of implementation and for interoperability with both SGML and HTML."

The XML 1.0 specification includes an excellent list of design principles including the following: XML documents should be human-legible and reasonably clear; also, the design of XML shall be formal and concise (Kay).

There are a number of other design goals that the XML Working Group was interested in achieving when designing XML. The design goals, as they are listed in the specification, are as follows:

XML shall be straightforwardly usable over the internet: The designers wanted XML to become the major interoperability standard, for data exchanged over the Internet, as well as within enterprises. They wanted XML to transparently interact with HTTP and other communication protocols that were already in place (Navarro, White, and Burman).

XML shall support a wide variety of applications: The designers of XML wanted to encourage the development of all kinds of tools that would create and interact with XML (Navarro, White, and Burman).

XML shall be compatible with SGML: This was difficult to achieve because creating a true subset means that all XML must also be valid SGML (Navarro, White, and Burman).

It shall be easy to write programs which process XML documents: The goal was to make the language small enough to enable the development of good tools in a reasonably short amount of time. Eventually the idea of a 'well-formed' XML document

was born to hasten the development of tools. This, in turn, would speed up the adoption of XML (Navarro, White, and Burman).

The number of optional features in XML is to be kept to the absolute minimum, ideally zero: The designers believed that options, although they make the language more flexible and powerful, can also prove troublesome and interfere with data exchange. The goal was to ensure that any XML processor could read any XML document (Navarro, White, and Burman).

The XML design should be prepared quickly: The group felt that the time was right to launch XML into the world and did not want to spend too much time developing the standard, or else the effort might get sidetracked (Navarro, White, and Burman).

XML documents shall be easy to create: The XML Working Group wanted to encourage the development of a wide variety of creation tools for all types of users. They did not want to limit users to either very specialized tools or plain text editors (Navarro, White, and Burman).

Terseness in XML markup is of minimal importance: The W3C Working Group chose clarity over cleverness, making all XML markup very explicit (Navarro, White, and Burman).

There were also two other goals which did not appear in the formal specification. The first had to do with internationalization, which resulted in the Unicode mandate. This was an aspect that was considered critical from the very beginning. The second was that the design of the language should be cleanly structured, enough so that someone could apply an easy scripting facility (Navarro, White, and Burman).

XML: A Family of Standards

XML is not made up of one technology; rather it is made up of a group of technologies. These technologies are referred to as a 'family of standards.' The 'XML family' provides extensions and applications of XML that build bridges to other formats, as well as making it easier to work with data ("XML in 10 points"). This family consists of the following specifications: XML Linking Language (including XLink, XPath, and XPointer), CSS, XSL (including XSLT and XSL-FO), DOM, XML Namespaces and XML Schemas 1 and 2.

The XML Linking Language (XLink) allows elements to be inserted into XML documents in order to create and describe links between resources ("XLink Version 1.0"). It uses XML syntax to create structures to describe both the simple unidirectional hyperlinks of today's HTML as well as more sophisticated multi-ended and typed links. Overall, XLink defines the relationship between two or more data objects as opposed to a whole document (Navarro, White, and Burman).

The XML Path Language (XPath) is a language used for addressing parts of an XML document rather than the whole document. It was designed to be used by both XSLT and XPointer ("XPath"). XPointer builds on XPath to support addressing into the internal structures of XML documents. XPointer allows for examination of a hierarchical document structure and choice of its internal parts based on various properties, such as element types, attribute values, character content, and relative position ("XPointer Version 1.0").

Cascading Style Sheets, Level 2 (CSS2) is a style sheet language that allows authors and users to attach style (i.e. fonts, spacing, and aural cues) to structured

documents such as HTML documents and XML applications. CSS2 builds on Cascading Style Sheets, Level 1 (CSS1) with very few exceptions ("CSS, Level 2"). The Extensible Stylesheet Language (XSL) is an advanced language for expressing stylesheets. It consists of two parts: a language for transforming XML documents, XSL Transformations (XSLT), and an XML vocabulary for specifying formatting semantics, XSL Formatting Objects (XSL-FO) ("XSL").

The Document Object Model (DOM) is a standard platform- and language-neutral application interface for manipulating HTML and XML data. It allows the user to access and update the content, structure, and style of documents. Therefore, a software program can write to the DOM rather than having to manipulate the XML directly (Navarro, White, and Burman).

XML Namespaces provide a way of assigning unique names to document constructs so that software can operate correctly and avoid collisions. The use of XML namespaces allow context to be given to element names, which allows them to remain unique and thus processable. In addition, namespaces allow markup from different XML applications to be used in the same document without conflicting (Navarro, White, and Burman).

XML Schemas 1 and 2 help developers to precisely define the structures of their own XML-based formats ("XML in 10 points"). XML Schema Part 1: Structures specifies the XML Schema definition language, which offers facilities for describing the structure and constraining the contents of XML 1.0 documents. XML Schema Part 2: Datatypes defines facilities for defining datatypes to be used in XML Schemas as well as other XML specifications ("W3C XML Schema").

Over time, it became apparent that XML 1.0, XPath, the XML Schema Language, and DOM all had similar but subtly different conceptual models of the structure of an XML document. Thus, the W3C XML Core Working Group began work on an XML Information Set (Harold and Means). The recommendation defines an abstract data set whose purpose is to provide a consistent set of definitions for use in other specifications that refer to the information in an XML document ("XML Information Set").

XML Standards Bodies

The International Organization for Standardization (ISO) is one of several organizations that have been involved in the development of XML standards. The ISO was created in 1946 by delegates from 25 countries as an international organization, of which the object would be "to facilitate the international coordination and unification of industrial standards" ("ISO Homepage"). Over the last several years, the ISO has grown to become a worldwide federation of national standards bodies from approximately 130 countries. The current mission of the ISO "is to promote the development of standardization and related activities in the world with a view to facilitating the international exchange of goods and services, and to developing cooperation in the spheres of intellectual, scientific, technological, and economic activity" (White, Quin, and Burman). Up to now, the ISO has been extremely successful in developing more than 14,000 International Standards ("ISO Homepage").

Another organization that has been involved in the development of XML standards is the Organization for the Advancement of Structured Information Standards (OASIS). OASIS was founded in 1993 under the name SGML Open as a vendor-only

consortium devoted to developing guidelines for interoperability among products that support SGML. Over time, OASIS added user members to contribute to the technical work of the consortium. As the consortium evolved, it made sense to accommodate related technologies such as CGM (Computer Graphics Metafile) and XML (Navarro, White, and Burman). OASIS eventually changed its name in 1998 to reflect the expanded scope of technical work, including SGML, XML, and other related standards ("OASIS Mission").

OASIS currently has more than 3,000 participants representing over 600 organizations and individual members in 100 countries around the world. Members themselves set the OASIS technical agenda, using a lightweight, open process designed to promote industry consensus and unite disparate efforts. OASIS produces worldwide standards for security, Web services, conformance, business transactions, supply chain, public sector, and interoperability within and between marketplaces ("OASIS Mission").

The World Wide Web Consortium (W3C) is the most influential body on industry acceptance because of the large number of XML applications and vocabularies that it has released. The W3C was created to lead the World Wide Web to its full potential by developing common protocols that promote its evolution and ensure its interoperability. The W3C was founded by Tim Berners-Lee in October 1994 at the Massachusetts Institute of Technology, Laboratory for Computer Science in collaboration with the Conseil European pour la Researche Nucleaire (CERN), also known as the European Laboratory for Particle Physics, with support from the U.S. Defense Advanced Research Project Agency (DARPA) and the European Commission. In April 1995, the Institut

National de Recherche en Informatique et Automatique (INRIA) became the first European W3C host, followed by the Keio University of Japan in Asia in 1996. Just recently, in 2003, the European Research Consortium in Informatics and Mathematics (ERCIM) took over the role of European W3C Host from the INRIA ("About the W3C").

The W3C consists of approximately 400 member organizations from around the world, all of whom have a vested interest in the Web. The W3C has committed itself to leading the technical revolution of the Web. Their long term goals, which reflect their mission, are the following: to make the Web accessible to all by promoting technologies that take into account the differences of culture, language, education, ability, material resources, access devices and physical limitations of users throughout the world; develop a software environment that permits each user to make the best of resources found on the Web; and guide the Web's development with careful consideration for the legal, commercial and social issues raised by this technology ("About the W3C").

The ISO, OASIS, and the W3C are three organizations that have been heavily involved in the development of XML standards. However, other key technologies have been developed by various standards granting bodies. They include some of the following: the Internet Engineering Task Force (IETF), the European Commission's Open Information Interchange (OII), and the European Computer Manufactures Association (ECMA) (White, Quin, and Burman).

Some industry analysts believe that there are too many web service standards bodies. However, officials directly involved within the organizations feel otherwise. Tom Glover, president and chairman of WS-I (Web Services Interoperability Organization), states that "it would be ideal if we had a single organization dealing with

Web service standards, but I think it's a practical impossibility." "Different organizations focus on different problems," he stressed (Krill). Glover suggests using the strengths of each organization to get the job done together. OASIS president and CEO Patrick Gannon added his sentiments, arguing that there is not one organization that governs every aspect of development in a set of standards. "I think what we have here is an opportunity for cooperation among different organizations," he said (Krill), and many seem to agree.

W3C Standards Process

Every recommendation that goes through the W3C standards body must endure a long and tortuous process of proposals and revisions before it's finally ratified by the organization's Advisory Committee. A recommendation typically begins when one or more W3C member companies submit a proposal of new technology. If it is properly submitted in HTML, according to the template provided, and is about an appropriate topic, the W3C liaison must acknowledge receipt of the submission. The acknowledgment generally contains some commentary about the submission, which could be either positive or negative. After the acknowledgment, the documents in the submission may be published by the W3C as a note (Navarro, White, and Burman).

A note can be the submission of a new technology. It can also be a discussion written by an active W3C Working Group for the clarification and integration of existing standards. However, a note is merely a dated, public record of an idea, comment, or document. It does not imply a commitment on the part of the W3C (Navarro, White, and Burman).

If the W3C decides that a submission is critical to its mission of creating interoperable standards across the Web, it creates a W3C Working Group to study the submission and other related topics. The working group then creates W3C Working Drafts of its work in a particular area (Navarro, White, and Burman). A working draft does not necessarily represent a finished work or consensus among the members; rather it represents a progress report on the project (Ray).

At this point in the process, developers begin implementing parts of the proposed technology to test it out, finding problems in the process. Software vendors also press for more features. This feedback is important to ensure that work is going in the right direction and nothing important is being left out (Ray).

When all the members of the working group have agreed that the current working draft meets the goals and requirements of the group, they issue a 'last call' for comments (Navarro, White, and Burman). The draft then becomes a candidate recommendation. At this point, the working group members are satisfied that the ideas are sound and no major changes will be needed. Experts continue to weigh in with their insights, addressing details and small mistakes. The deadline for comments arrives and the working group goes back to work, making revisions and changes to the working draft (Ray).

The working draft does not become a proposed recommendation until the director of the W3C Advisory Committee proposes it to the advisory committee for review. Then, all of the members of the advisory committee must reach a consensus on the proposed recommendation before it can become a recommendation. After the advisory committee completes their vote, it is then up to the director whether or not it

becomes an actual recommendation. After the director makes a decision, the following may happen to the working draft: it is issued as a recommendation, issued as a recommendation with minor changes indicated, returned to the working group for modification, or abandoned and removed from the W3C agenda (Navarro, White, and Burman).

XML was developed according to normal W3C practice. It was developed by a small group of people who received technical input from a larger Special Interest Group, which in turn drove the editorial processes that led to the creation of the XML specification. During the standards process, the specification progressed through a set of design goals and succession of interim drafts, which eventually resulted in the XML 1.0 Recommendation, issued by the W3C in February of 1998 ("XML Fact Sheet").

Key Features of XML

There are a number of key features of XML. These key features help distinguish XML from other programming languages. They include the following:

XML is optimized for use on the Internet: XML is interoperable and carefully designed to avoid the requirement for the delivery of multiple document components when one is sufficient. Also, all external addressing within the XML domain is accomplished through the use of standard Web addresses ("XML Fact Sheet").

XML is built on the experience of SGML: XML, which is much simpler than SGML, and optimized for network applications, is fully compatible; thus, leveraging the substantial installed base of SGML tools and experience ("XML Fact Sheet").

XML is easy to process: Programs which process XML are easy to write. The number of implementations that can be found on the Internet is well into the double digits, and is rapidly growing ("XML Fact Sheet").

XML has a solid base for internationalization: XML draws on several generations of internationalization experience. It also avoids the pitfalls of insufficient attention to internationalization, and of being so general as to impair interoperability. This is possible because of the use of the Unicode (ISO 10646) standard for internationalized character sets ("XML Fact Sheet").

XML is a general-purpose tool: The design of XML includes many features designed to support authoring, indexing, and other types of applications ("XML Fact Sheet").

XML is designed to support automation: The XML specification includes a precise and rigorous set of rules for error and exception handling, which is unlike any other Internet data format. This empowers application builders, who create XML software using relatively lightweight and simple data-handling modules, in the exception that the data will usually be well-formed, and that when an error occurs, the correct fallback procedures are well-known and common across the industry ("XML Fact Sheet").

XML is self-describing: No prior knowledge of an XML application is needed (Tittel, Pitts, and Boumphrey).

XML documents are well formed: XML documents must follow certain rules, which make such documents easier to read and create (Tittel, Pitts, and Boumphrey).

XML is for structuring data: XML includes a set of rules for designing text formats that let the user structure their data (i.e. structured data includes spreadsheets, configuration parameters, and financial transactions). XML makes it easy for a computer to generate

data, read data, and also ensure that the data structure is unambiguous ("XML in 10 points").

XML is modular: XML allows the user to define a new document format by combining and reusing various other formats. Formats that are developed independently of one another may have attributes or elements with the same name. To eliminate confusion when combining these formats, XML provides a namespace mechanism. XSL and RDF are some examples of XML-based formats that use namespaces ("XML in 10 points").

XML is license-free, platform-independent and well-supported: There is a large and growing community of tools and engineers that are experienced in XML. Since XML is license-free, the user can build their software around it without having to pay anything. Also, the growing support for XML means that the user is not tied to a single vendor ("XML in 10 points").

W3C Role in XML Development

XML was developed by the W3C in 1996 in response to a concern by many individuals, as stated earlier, regarding a lack of flexibility in describing documents and data found in HTML. It was developed by an XML Working Group (originally known as the SGML Editorial Review Board) that was formed under the supervision of the World Wide Web Consortium (W3C). The XML Working Group was chaired by Jon Bosak of Sun Microsystems with the active participation of an XML Special Interest Group (previously known as the SGML Working Group), also organized by the W3C. In addition, Dan Connolly served as the Working Group's contact with the W3C ("Extensible Markup Language").

The XML 1.0 specification followed the normal standards process. Along the way, five different working drafts were released. In December of 1997, the W3C XML Working Group had determined that the XML 1.0 specification was stable, contributed to Web interoperability, and was supported for industry-wide adoption; thus, it was released as a Proposed Recommendation for review and voting by the W3C member organizations. By February of 1998, it had received Recommendation status by the W3C ("Media Alert: XML 1.0 as Proposed Recommendation").

Following the release of the XML 1.0 Recommendation, the W3C developed several other recommendations pertaining to XML. They include the following: **XLink**, a method of creating and describing hyperlinks; **XML Base**, a method for defining base URIs for parts of XML documents; **XPointer**, a language which builds on XPath to support addressing into the internal structures of XML documents; **XPath**, a language used to address, point to, or match portions of XML documents; and **XML Schema**, a language that provides a means for defining the structure, content and semantics of XML documents. More recent specifications include **XSL**, an advanced language for expressing stylesheets; **XSLT**, a language for transforming XML documents into other XML documents; and **XQuery**, a highly malleable combination of an SQL-like syntax joined with XPath.

The W3C Core Working Group has continued to develop and maintain the specifications for XML and other closely related specifications. The W3C has also become the primary center for developing other cross-industry specifications that are based on XML. Some of these specifications are being developed within the XML

Activity, including XML Query and XML Schema, and some are being developed in other W3C Activities, such as DOM, SVG and XHTML ("Extensible Markup Language").

Corporations' Role in XML Development

XML was created and developed by the W3C XML Working Group, which includes many key industry players such as Adobe, ArborText, DataChannel, Inso, Hewlett-Packard, Isogen, Microsoft, NCSA, Netscape Communications, SoftQuad, Sun Microsystems, Texcel, Vignette, and Fugi Xerox, along with experts in structured documents and electronic publishing. "The commitment of strong competitors such as Sun, HP, Microsoft, and Netscape to work together on an open standard for information exchange has been a remarkable demonstration of cooperation for the common good," said Jon Bosak, Sun's Online Information Technology Architect and Chair of the W3C XML Working Group. "XML represents a key technical advance in web technology," he added ("Press Release: XML 1.0 as Recommendation").

Many of the companies involved in the creation and development of XML have continued to contribute their expertise by involving themselves in XML related initiatives. For example, Sun Microsystems is actively participating in W3C working groups for XML Stylesheet/Transformation Language (XSL/T), XML Schema, XLink, and XML Query. Sun is also involved in a number of other industry consortia including OASIS, XML.org, and Apache ("XML at Sun: FAQs").

Other companies, such as Microsoft, have forged ahead creating a path which many have followed. From the beginning, Microsoft has been actively involved in defining the XML standard. In 1998, they became the first major software vendor

whose browser incorporated support for the emerging XML specifications developed by the W3C. Microsoft Internet Explorer 5 and the Windows operating system incorporated the latest XML technologies including XML 1.0, XSL, XML DOM, and XML Namespaces ("Industry's Most Complete Implementation of XML"). Since 1998, Microsoft has built XML support into most of its products including Microsoft SQL Server, Microsoft Exchange, Microsoft BizTalk Server, Microsoft Office, the Microsoft .NET Framework, and Microsoft Visual Studio .NET (International).

Over the past several years, Microsoft has also continued rigorous open standardization of XML and Web Service technologies. In 2001, Microsoft, along with key industry players such as IBM, DataChannel, Hewlett-Packard, Intel, Oracle, webMethods, and many others submitted the Web Services Description Language (WSDL) specification in hopes of standardizing it ("Microsoft Co-Submits Another Web Services Spec. to W3C"). As of March 2004, WSDL Version 2.0 Part 1: Core Language was in the working draft stage of the W3C standards process.

As XML has gained increased popularity, it has been and is being integrated into software products from nearly every major company, including Adobe Systems, Oracle, and IBM. Adobe recently incorporated support for XML into its ever popular Acrobat family of software and FrameMaker software. In 2001, Oracle integrated XML into its entire product stack and made numerous product enhancements to support XML, including improvements to Oracle Tools and Utilities, Oracle9*i* Database, and Oracle9*i* Application Server software (Hall). In 2002, IBM incorporated support for XML into its WebSphere Software, Lotus Notes and Domino 6; and in 2003, they unveiled new

software called DB2 Information Integrator. The software provides a consolidated view of data scattered across multiple data sources throughout a company.

Government's Role in XML Development

Decision makers at every level of state and local governments have wrestled with the question of how to transfer information between systems. HTML was considered; unfortunately, it cannot be the complete solution because it does not have the ability to 'tag' or label an object on one site so that another site or system can recognize the same object. However, XML provides a mechanism to label sets of data so that they can be shared between systems and among different agencies (NECC).

The United States federal government's XML Working Group was created by the Chief Information Officers Council (CIOC) in June of 2000. Its goals were to partner with standards bodies that are developing XML and facilitate the government's broad transition from electronic data interchange (EDI) to XML-based exchanges. At the time at which the working group was created, the U.S. Navy had already written a set of XML guidelines to cover its operations; thus, the government-wide work group took the Navy's document and generalized it (Kotok).

It is typical that organizations as large as the U.S. federal government rarely move quickly, so it is surprising to see the amount of XML activity currently underway. However, many government agencies are not strangers to markup. The Navy embraced XML's ancestor SGML when it was released in the 1980's. The Navy also contributed to the development of numerous other standards such as the CALS tables in DocBook and the HyTime standard (Ford).

The increased awareness of XML within government organizations is in part because it is required by law. The OMB Circular A-119 and the National Technology and Transfer Act of 1995 require that federal agencies first use VCSs, or Voluntary Consensus Standards to carry out policy objectives. These VCSs are typically XML-based schemas that are based on standards established by the W3C, OASIS, and similar standards organizations (Ford).

In addition, the Government Paperwork Elimination Act of 1998 requires federal agencies to use electronic documents and accept electronic signatures as of October, 2003. This represents another potential use of XML. Taking all of this into account, not only is there historical pressure to use XML, which comes from the use of markup in organizations like the U.S. Department of Defense and the U.S. Internal Revenue Service; there is also a legal requirement (Ford).

The recent deadline for the Government Paperwork Elimination Act was not met by some agencies; however, others went beyond what was required to create better business practices. The U.S. Air Force is one agency that used the compliance effort as an opportunity to improve their business processes. In October 2002, they started converting paper and static Web forms into interactive electronic forms. By June of 2003, the Air Force had converted approximately 18,000 XML-based E-forms and was deemed compliant with the act (Kontzer).

The Internal Revenue Service is another government agency that has embraced the idea of E-documents, even though it is one of the few agencies that were exempt from the Government Paperwork Elimination Act. More than 150 IRS forms have been configured for electronic filing; and in 2002, approximately 53 million taxpayers

submitted their 1040 returns electronically. The agency is continuing to add additional XML-based E-forms to aid taxpayers in the future (Kontzer).

Overall, the U.S. government has been wary of the government-wide adoption of XML. In April of 2002, the General Accounting Office (GAO) published a document entitled "Challenges to Effective Adoption of the Extensible Markup Language". The document identified a number of problems with XML, including the lack of a central registry, the risk of redundant schemas, and the lack of security. However, they recommended that the U.S. government develop a strategy for the government-wide adoption of XML, which would ensure the use of this technology across agencies (Ford).

Currently, within the U.S. government, centralization is the exception, not the norm. There are numerous XML applications that are scattered across different government agencies with the Department of Defense, Environmental Protection Agency, Internal Revenue Service, and others creating XML schemas as needed and applying them internally. In an effort to encourage the centralization of all online government services, including those that use XML, the White House has created the E-government initiative (Ford).

As XML is increasingly used throughout government, there continues to be numerous attempts to centralize documents, data sets, and schemas to eliminate redundancy. One such attempt is the XML.gov registry which was originally created by the XML Working Group as a government-wide registry and now serves as the government's XML portal. The Web-services.gov site provides users insight into how the government might use XML as a data transmission language instead of solely as a

document markup language. These both constitute the first steps towards centralization of all online government services (Ford).

CHAPTER III:

THE CURRENT STATUS OF XML

Success of XML

Many experts believe that the real reason for XML's overall success and widespread use is because of its accessibility. There are two main reasons for XML's availability to the general public. First of all, the XML specification is simple to read and easy to understand. Secondly, the tools to work with XML are readily available to ordinary programmers. In addition, the XML tools that have been developed go beyond mere parsing and have extended to transformation (Kay).

Widespread Use of XML

XML has been talked about for several years; however, it's now beginning to have an impact on the real world. XML can be seen everywhere, from page layout applications to databases, from peer-to-peer messaging systems to complex business-to-business data exchanges (White, Quin, and Burman). XML has been adopted in fields as diverse as law, aeronautics, finance, insurance, robotics, multimedia, art, travel, hospitality, telecommunications, software, agriculture, physics, journalism, theology, comics, and much more. XML has become the new syntax of choice for document formats and is used by almost all computer applications. It is used on Linux, Macintosh, Windows, and many other computer platforms (Harold and Means).

The overall acceptance of XML by the Internet community has opened the door for the development of an entire family of XML standards. Some of which include stylesheets for display and transformation, methods for linking resources, tools for data

manipulation and querying, error checking and structure enforcement tools, and a number of development environments. Many experts believe that because of these new applications, XML is expected to have a long and faithful career as the structured information toolkit of choice (Ray).

There are hundreds of XML applications that have been developed by the W3C and other standards bodies, including OASIS. There are even more informal, un-standardized applications that have been developed by individuals and corporations (Harold and Means). These applications are widely used for anything from chemical formulas to genealogies and the assortment of XML-based markup languages is likely to continue growing (Tittel, Pitts, and Boumphrey).

XML is used by a wide variety of users; however, typical use involves three main areas: documentation professionals and researchers, web developers, and database and object-oriented programming (OOP) developers. A number of people use XML to maintain documentation and cataloging for their companies. In addition, researchers use XML for a wide range of uses from the Human Genome Project to exchanging chemical and other data. XML is used by web developers in employing dynamic web site development. Database developers are using XML to extract data from databases and display it using server-side processing scripts such as JSP (JavaServer Pages), ASP (Active Server Pages), or Perl (White, Quin, and Burman).

Browser Support of XML

Modern web browsers do not yet support XML directly. However, a number of browsers continue to incorporate support for CSS or XSL stylesheets to be used to style

XML data for presentation. Microsoft's Internet Explorer 6 offers XML support including full support of Dynamic HTML (DHTML), CSS1, and DOM 1. It also provides some support for the Synchronized Multimedia Integration Language (SMIL) (Tittel, Pitts, and Boumphrey).

Netscape 4.x and earlier versions did not provide any significant support for displaying XML in the web browser. Both Mozilla and Netscape 6.0 fully support the display of XML in the web browser (Harold and Means). Mozilla also provides support for several W3C Recommendations and drafts from the XML family of specifications, including XHTML, XML Base, XLink, and XPointer, as well as other related technologies ("XML in Mozilla").

Amaya is the W3C browser and web editor which is used to demonstrate and test new developments in Web protocols and data formats. It supports XML and an increasing number of XML applications, including XHTML, MathML, and many CSS2 features. Amaya also includes SVG support (Tittel, Pitts, and Boumphrey).

Obstacles of XML

While many experts boast about how easy it is to integrate applications using XML, others feel that this is not the case. CRN Test Center engineers believe that achieving a pure XML solution is quite a difficult task. An increase in the number of XML standards has opened the door to a slew of solution providers who have begun to integrate and streamline enterprise systems using XML. However, the flurry of new technologies has created a complex web of access methods that have made their work largely inaccessible to mainstream developers. Microsoft and BEA Systems have tried

to alleviate the complexity by offering development environments that generate code and hide it. Nonetheless, direct access to XML is still a necessary evil when integrating applications (Morejon).

Another big integration hurdle for mainstream developers has been allowing applications to communicate by exchanging and translating XML documents. XML schemas lay at the heart of XML integration methods. Although, XML schemas and DTDs are used extensively and promoted by many vendors, they are quite complex and require expert knowledge of data mapping in order to fully understand them (Morejon).

The number of Business-to-Business (B2B) Web services success stories does not seem to compensate for the many more unhappy endings. Quite often these failures are the result of overestimating the capabilities of XML. A number of attempts have been made by industry consortia to create standardized languages or schemata; however, it is evident that no two companies communicate data in exactly the same way. Also, many companies deal with businesses that are outside of their industry (Rapoza).

An additional challenge of XML adoption is security. Initially, security was an obvious concern of many because XML adoption is driven by e-business initiatives. However, digital signature capability has been built into current XML standards so that XML documents sent across organizational boundaries can be authenticated.

Some experts believe that identity management and service authorization are the two biggest security challenges present in the shift from a traditional application security model to that which is needed for securing XML-based Web services. Many businesses have experience securing applications; however, securing services requires a whole

new set of tools. It is recommended that companies revamp their security review processes and clarify the security responsibilities of their business partners in order to move towards securing Web services (Bednarz, "XML security integration").

Progress has been made with relation to identity management standards. Liberty Alliance specifications and the Web Services Federation Language (WS-Federation) tackle taxonomy, roles, and responsibility of identity management. There are existing standards that address single sign-on at a basic level; however, the issue of identity re-establishment has not been resolved. Also, the ability for a user to delegate authority has not been addressed (Bednarz, "XML security integration").

Best Practices

The basic concepts of XML have been around for a long time, but have only recently been applied to computer data files. Over the centuries, the idea of extending markup from a paper tool to an electronic one progressed slowly. It was difficult to generalize the basic concepts of markup so that they could be used for anything made up of component parts, including non-physical objects (Phillips).

Generally, people organize almost everything into hierarchies. Any hierarchical structure can be described using XML; however, it does have its limitations. XML is not truly object-oriented. XML documents cannot inherit from their ancestors and they are not truly encapsulated because their internals are fully exposed. Nonetheless, they do exhibit polymorphism and other object oriented behaviors (Phillips).

There are a number of best practices that are currently discussed in the XML community. Selected best practices include:

Always Use Namespaces: Namespaces make it easier to assemble large schemata from smaller schemata because they prevent name conflicts. In document instances, elements should always be qualified with a namespace, either by using a default namespace or a namespace prefix (Daum and Merten).

Don't Reinvent the Wheel: If possible, use existing document types and extend these to your requirements. This will result in higher quality schema definitions and ensure that the core concepts of the XML document are understood by others (Daum and Merten).

The Use of Multipart Schemata: Some authors warn against using multipart schemata because they multiply the dependencies between software artifacts, ultimately making a system harder to maintain. However, others support the use of multipart schemata to construct larger schemata from smaller components and type libraries (Daum and Merten).

Avoid External Entities: External Entities increase the dependencies between software artifacts. Also, not all XML processors support external entities (Daum and Merten).

Never Change a Published Schema: Instead, create a new schema with each change (Daum and Merten).

Use Only Version-Controlled Schemata: When using a schema from a public repository, make sure that the repository has a version control system in place. Schemata that are not controlled by a version control system can change at any time (Daum and Merten).

Consider Equipping Each Document Element with a Universally Unique Identifier (UUID) Attribute: Doing so, helps to identify an element. One advantage to this is that elements can keep their identity even through transformations when merged with another document, or when moved to a different location (Daum and Merten).

Adopt a Concise Style for Schema Design: Write element and attribute names in camel case, usingInternalCapitalizationInLieuofWhiteSpaceLikeThis. Names should be meaningful and describe the marked item sufficiently. Also, elements and attributes should be named by their function, not their position in a set. Do not use a complex string expression within elements that requires custom parsing (Daum and Merten).

Do Not Use Exotic Language Elements: Only use language elements that are commonly supported by existing tools including XML parsers, editors and viewers (Daum and Merten).

Stay with XML 1.0: XML 1.1 does several things, one of which is marginally useful to a few developers and the rest are aggressively harmful (Harold, "Effective XML").

Always Use an XML Declaration: It is recommended that every XML document have an XML declaration because it helps human users as well as software identify the document as XML. It identifies the version of XML in use, specifies the character encoding, as well as helps in optimizing the parsing (Harold, "Effective XML").

Markup with ASCII if possible: These are the only characters that can be reliably displayed and edited across a wide range of computers and software that are in use today (Harold, "Effective XML").

Modularize DTDs: Divides a DTD into multiple, independent units of functionality that can eventually be combined into a single application (Harold, "Effective XML").

Make Structure Explicit through Markup: All structure within an XML document should be indicated using XML tags (Harold, "Effective XML").

A number of the best practices listed above should be implemented when working with XML in a global environment, along with the following:

Write in UTF-8 (Unicode Transformation Format, 8-bit encoding form): The default encoding for all XML documents is UTF-8, which is compressed encoding of Unicode (International).

Make sure XML data is locale- and culture-neutral: Using XML entails making sure that the data is cleanly separated from the user interfaces (International).

Industry-Specific XML Applications

Various industry consortiums are bringing together technology and professional service companies from around the world to fuel the adoption of XML-based vocabularies and streamline information exchange among companies belonging to a particular industry. These industry-specific standards are built upon the core XML technologies (Kim). They also provide powerful tools to display and work with XML documents.

There are a number of XML technologies that are dedicated to providing ways of displaying documents in different formats. They include some of the following: the Extensible Hypertext Markup Language (XHTML), the Wireless Markup Language (WML), Scalable Vector Graphics (SVG), and the Synchronized Multimedia Integration Language (SMIL) (Kim). In addition, the Resource Description Framework (RDF) and the XML-Data specification are two XML-based technologies that provide a way to reference and manage data from disparate sources and among various formats.

XHTML is a reformulation of HTML 4.0 document types as applications of XML 1.0. It was published as a W3C Recommendation in January of 2000. However, a revised Recommendation was released in August of 2002 (Kim).

WML is an XML vocabulary that is used for the fast delivery of information and services to wireless users (Kim). It was designed specifically to meet the constraints found in wireless devices. Some of these constraints include a small display area with limited user input capabilities, limited bandwidth, and limited CPU power and memory space (Tittel, Pitts, and Boumphrey).

SVG is a language used to describe two-dimensional graphics in XML. It allows for three types of graphic objects: vector graphic shapes, images, and text. These graphical objects can be grouped, styled, transformed, and easily embedded into XML documents. SVG was published as a W3C Recommendation in January of 2003 ("SVG 1.1").

SMIL enables the integration of a set of independent multimedia objects into a single synchronized multimedia presentation. It is typically used for media rich presentations that include streaming audio and video with images, text or any other media type. SMIL documents are XML documents that can be created with a simple text editor. SMIL 2.0 received W3C Recommendation status in August of 2001 ("W3C Synchronized Media Home page").

RDF is an XML-based language for representing metadata about Web resources, such as the title, author, and modification date of a web page, or the copyright information of a web document ("RDF Primer"). It assures interoperability between applications that exchange application- or platform-specific data across the Web. In general, RDF provides the basis for generic tools used for authoring, manipulating, and searching machine-readable data on the Web (Tittel, Pitts, and Boumphrey). RDF became a W3C Recommendation in February of 1999. A revised

Recommendation that was intended to replace the original specification was recently released in February of 2004.

XML-Data is an XML vocabulary that provides a mechanism to reference binary data within XML documents. XML-Data represents an XML element that allows the user to describe text structures, relational schemas, and more. In other words, at the center of XML-Data is a DTD for describing DTDs (Tittel, Pitts, and Boumphrey).

A number of scientific concepts are known to be universal. Unfortunately, there exists the lack of a truly universal and efficient means of electronically expressing complicated formulas. The Chemical Markup Language (CML) and the Mathematical Markup Language (MathML) help to address this need. CML is an XML vocabulary used for representing molecular information, covering macromolecular sequences to inorganic molecules and quantum chemistry (Kim). MathML is an XML application used for describing mathematical notation, capturing both its structure and content. It enables mathematical expressions to easily be sent, received, and processed on the Web; similar to how HTML has enabled this functionality for text. MathML Version 2.0 was published as a W3C Recommendation in February of 2001. Recently, a second edition of MathML 2.0 was released as a W3C Recommendation in October of 2003 ("MathML 2.0 (Second Edition)").

The publishing industry is responsible for creating a large amount of useful content on a daily basis. The News Markup Language (NewsML) and DocBook are two standards which enable loosely structured documents to be expressed as XML documents. NewsML is an XML standard that is used for describing news articles for use in print or web publishing (Kim). DocBook is an XML application that is designed to

capture computer documentation and other types of lengthy, complex documents such as books and articles. It is currently being used worldwide by hundreds of organizations that manage millions of pages of documentation in a variety of print and online formats (Tittel, Pitts, and Boumphrey).

Over the past several years, the financial industry has benefited from advances in Internet technologies. They are now eagerly pursuing the development of XML based standards for describing business processes. Two of the financial industry's most widely adopted standards are the Extensible Business Reporting Language (XBRL) and the Financial Products Markup Language (FpML) (Kim).

XBRL is an XML-based standard that is used for identifying and improving the communication of complex financial data among public and private companies. It also makes the analysis and exchange of financial information easier and more reliable. XBRL is used by individuals in the accounting profession, regulators, analysts, the investment community, capital markets, and lenders ("What is XBRL").

FpML is the business information exchange standard for the electronic dealing and processing of financial, "over-the-counter" derivative instruments. It establishes a protocol for sharing information on, and dealing with, swaps, derivatives, and structured products. FpML is primarily used to automate the flow of information across financial institutions ("What is FpML?").

XML technologies are currently being employed to solve today's medical challenges. Two of the relevant medical standards are the Gene Expression Markup Language (GEML) and Health Level 7(HL7). GEML is an XML vocabulary that is used for biological gene expression data, using bioinformatics professionals to catalog and

search for genetic data. HL7 is a protocol for electronic data exchange among healthcare information systems. It is used extensively by the healthcare industry, including pharmacy, medical devices, imaging, insurance claims processing, and clinical and administrative data. Its primary purpose is to improve patient care and ensure the interoperability between healthcare information systems (Kim).

Recently Developed XML-based Languages

XML Query (XQuery)

The ability to query XML data has become increasingly important. XML is able to represent many different types of information from diverse sources. Thus, an XML query language must provide features for retrieving and interpreting information from these diverse sources ("W3C XML Query").

XQuery was designed with three goals in mind: to produce a data model for XML documents, design a set of query operators on that data model, and to create a query language based on these query operators. XQuery is designed to be a language in which queries are concise and easily understood. It is also flexible enough to query a broad range of information sources including databases and documents ("W3C XML Query").

The overall design of XQuery is based on an XML query language called Quilt. Quilt was influenced by the functional approach of the Object Query Language (OQL), along with the keyword-based syntax of the Structured Query Language (SQL), and other XML query languages including XQL, XML-QL, and Lorel. XQuery is also closely related to XPath and is therefore defined as a superset of XPath (Chamberlin).

XQuery is a functional language, which means that it is made up of expressions that return values. There are several kinds of expressions, many of which are composed of lower-level expressions combined by operators, symbols, or keywords (Chamberlin). XQuery is also a strongly-typed language which means that the operands of various expressions, operators, and functions must conform to the expected types ("W3C XML Query").

XQuery is currently in the Working Draft stage of the W3C process and is continuing to evolve. The XQuery syntax has received some criticism for not being in XML; however its syntax was developed for human-readability. An XML version of the XQuery syntax, XQueryX, is also in the Working Draft stage of the W3C process, and because it is XML has the potential for success as an alternative syntax (Daconta, Obrst, and Smith).

XQuery has earned the reputation by many as being one of the most cautiously developed and thus, slowest evolving W3C standard. One reason for this is because there is little industry experience in the retrieval and information storage of XML. There are a number of companies that are still innovating in this field and are generating a good deal of information that needs to be processed before a level of comfort with the XML query language can be achieved (Ivanov).

Even though XQuery is still not a W3C Recommendation, there have been some early adopters of the language. Key market players such as Lucent Technologies, Software AG, Microsoft, and Oracle have released products based on the working drafts. The delay in the final release of XQuery is one reason why there have not been large scale marketing campaigns of the implementations (Ivanov). However,

many analysts believe that as XQuery evolves and is adopted in the industry, its future will begin to look promising.

XML Key Management Specification (XKMS) Version 2.0

Public Key Infrastructure (PKI) is a well suited technology for its use in securing web services, however PKI deployment is too cumbersome and costly for it to achieve wide spread use. XKMS aims to reduce the costs of PKI without sacrificing its benefits. Essentially, XKMS borrows the best of PKI without reducing scalability or security. It creates a trust service that shields clients from its complexity by providing an XML interface to PKI (Salz).

With XKMS, a client and application server share an XKMS service in order to validate each other and process requests between the two of them. XKMS replaces several PKI protocols and data formats with one XML-based protocol. The XKMS protocol provides three fundamental operations:

1. Locating, which retrieves a cryptographic key to communicate securely with another entity
2. Validating, which makes sure the key is active and has not been revoked
3. Registering, which issues, reissues, and revokes keys

With XKMS, trust decisions are given to a common server so that they are centralized and can be applied consistently across platforms (Salz).

XKMS Version 2.0 was recently released as a Candidate Recommendation on April 4, 2004. It addressed the Last Call issues that were related to the April 18, 2003 W3C Working Draft and was loosely based on the March 30, 2001 W3C Note on XKMS

("W3C XML Key Management"). XKMS is made up of two parts: the XML Key Information Service Specification (X-KISS) and the XML Key Registration Service Specification (X-KRSS). X-KISS is a protocol designed to support the delegation by an application to a service of the processing of key information associated with XML signature, XML encryption, or other public keys. X-KRSS is a protocol designed to support the registration of a key pair by a key pair holder, with the intent that the key pair will subsequently be usable in conjunction with X-KISS or PKI ("W3C XML Key Management").

XML-based public key management was designed to meet two goals. The first is to support a client's ability to make use of sophisticated key management functionality. The second is to provide public key management support to XML applications that are consistent with the XML architectural approach. It is also important that XML key management be able to support the public key management requirements of XML Encryption, XML Digital Signature, and to be consistent with the Security Assertion Markup Language (SAML) ("W3C XML Key Management").

VoiceXML 2.0

The origins of VoiceXML began in 1995 with the development of an XML-based dialog design language that was intended to simplify the speech recognition application development process within an AT&T project called Phone Markup Language (PML). As AT&T began to reorganize, teams at AT&T, Motorola, and Lucent Technologies continued working on their own PML-like languages. By 1998, AT&T and Lucent had created different variants of their original PML. Motorola had developed VoxML, and

IBM was developing its own SpeechML. HP and PipeBeach had also developed similar languages for dialog design ("VoiceXML Version 2.0").

Soon after, the VoiceXML Forum was formed by AT&T, IBM, Lucent, and Motorola as a way to pool their efforts. Their mission was to define a standard dialog design language that developers could use to build conversational applications. They chose to use XML as their basis for this effort because they believed that that was the direction technology was going ("VoiceXML Version 2.0").

In 2000, the VoiceXML Forum released VoiceXML Version 1.0 to the public; shortly thereafter, it was submitted to the W3C for consideration as a new international standard. VoiceXML 2.0 is the result of work done to VoiceXML 1.0 based on input from W3C Member companies, other W3C Working Groups, and the public. VoiceXML 2.0 was recently approved as a W3C Recommendation in March of 2004 ("VoiceXML Version 2.0").

VoiceXML 2.0 is used for creating audio dialogs that feature synthesized speech, digitalized audio, and recognition of spoken and dual-tone modulated frequency (DTMF) key input. It is also used for the recording of spoken input, telephony, and mixed initiative conversations. The major goal of VoiceXML is to apply the advantages of Web-based development and content delivery to interactive voice response applications ("VoiceXML Version 2.0").

VoiceXML 2.0 is part of a handful of specifications in the W3C's evolving Speech Interface Framework. Other elements of the framework include the supporting Speech Recognition Grammar Specification (SRGS), used to describe the words and phrases that end users are expected to give in response to spoken prompts. Also included in

the framework are: Speech Synthesis Markup Language (SSML), which is used to create spoken prompts; Voice Browser Call Control (CCXML), which provides telephony call-control support for VoiceXML and other dialog systems; and Semantic Interpretation for Speech Recognition, which defines links between grammar rules and application semantics so that an application recognizes two spoken variations of the same element (Bednarz, "New speech technologies").

VoiceXML has become broadly adopted. It is the standard scripting language for making Web content accessible via voice and phone. VoiceXML's acceptance by the development community is reflective in the number of software vendors that have created VoiceXML 2.0-compliant applications, products and services, including HP, IBM, Lucent, and Motorola (Bednarz, "New speech technologies").

XML Inclusions (XInclude)

A number of programming languages provide some kind of inclusion mechanism that facilitates a modular storage of parts of a program. XInclude provides a processing model and the syntax for general-purpose inclusion of XML data. Inclusion is accomplished by merging multiple XML information sets into a single composite infoset (Birbeck, et al). Essentially, XInclude provides the means for building a single XML document out of multiple well-formed XML documents.

The specification of XML documents (infosets) to be merged and control over the merging process is done using elements and attributes written in the XML 1.0 syntax. The additional use of XML Namespaces ensures unique names for this special markup.

XInclude is a low-level process that is intended to make the resulting information set available to higher level applications (Birbeck, et al).

XInclude was recently released as a Candidate Recommendation on April 13, 2004. It addressed the Last Call issues that were related to the November 11, 2003 W3C Working Draft. The syntax defined in the specification leverages existing XML constructs such as elements, attributes, and URI references ("XInclude Version 1.0").

Updates and Modifications to XML standard

The XML standard recently received some updates and modifications to the prior version 1.0, which was released as a W3C Recommendation in 1998. The overall philosophy of names has changed since XML 1.0; whereas XML 1.0 provided a rigid definition of names and everything that was not permitted was forbidden. In XML 1.1, names are designed so that everything that is not forbidden is permitted. Also, since Unicode has and will continue to grow past version 4.0, further changes to XML will be avoided by allowing almost any character, including those not yet assigned names ("Extensible Markup Language").

XML 1.1 was also designed to correct a problem with Unicode line ending rules. XML 1.0 attempts to adapt to the line-end conventions of various modern operating systems; however, it discriminates against the conventions used by IBM and IBM-compatible mainframes. In order to allow straightforward interoperability between mainframe and non-mainframe systems, XML 1.1 added NEL to the list of line-end characters ("Extensible Markup Language").

There is a considerable demand to define a standard representation of arbitrary Unicode characters in XML documents. Therefore, XML 1.1 allows the use of character references to control certain characters that are forbidden in XML 1.0. Finally, XML 1.1 defines a set of constraints called 'full normalization' on XML documents, which document creators are strongly encouraged to adhere to, and which document processors should verify ("Extensible Markup Language").

A new XML version, rather than a set of errata to XML 1.0, was created because the changes affected the definition of well-formed documents. There have been modifications made with respect to what is a well-formed document and what is not. The XML 1.1 document was released as a W3C Recommendation on February 4, 2004 and was edited in place on April 15, 2004 ("Extensible Markup Language").

CHAPTER IV:

THE FUTURE DIRECTION OF XML

Future Research and Investigation of XML

There has been an ongoing effort within the W3C XML Core Working Group to republish the IETF MIME specification for XML, RFC 3023. There has also been work done on advancing the XInclude specification again after it was sent back to the Working Draft stage in 2003. The XML Core Working Group will continue to develop and maintain the specifications for XML and other related specifications ("Extensible Markup Language").

The W3C recently decided to take on the task of analyzing the issues surrounding the exchange of XML information in a binary format. In September of 2003, the W3C held a public workshop to address the issue of whether the W3C should develop a binary interchange format for people needing greater efficiency than what was claimed possible using text. The conclusion of the workshop was that it was not clear whether the benefits of such a format would outweigh the costs. They determined that the issues involved needed to be analyzed further. The XML Binary Characterization Working Group was formed in March of 2004 to investigate these issues, and to put in place precise measurements to be made of the benefits of binary interchange format over the existing methods of textual interchange ("Extensible Markup Language").

The XML Query Working Group has been moving closer to getting the XML Query 1.0 specification released as a Recommendation by trying to resolve as many open issues as possible. However, because of a high volume of comments from the

first Last Call, a second Last Call for the specification is expected later in 2004. Right now, a major focus of the XML Activity is to maintain a connection between Working Groups seeing that a large number of important specifications are being prepared and perfected as they progress towards Recommendation status ("Extensible Markup Language").

Until recently, the 'corporate database' has been untouched by XML. The requirement for a database technology that can handle the full spectrum of highly-structured information to highly-unstructured information is fairly obvious, but the goal remains elusive. The basic technology components are in place. However, a number of vendors have conflicting views on the best way to build a database that holds XML (Kay).

The challenge that currently exists for the industry is how to use the technology when it becomes available. There are two competing approaches. One is to treat an XML database as if it were a filing cabinet in which to store XML documents. The second approach is to design the database as a store of information, not just a store of XML documents. With either approach, the future of XML databases remains to be seen (Kay).

Problems with XML to be addressed

Some experts believe that the W3C has lost interest in fixing some of the problems at the core of XML. One unaddressed issue is the issue of identifying a document, its stylesheet, accompanying resources, and schemas as a 'package' rather than a mixture of processing instructions and attributes. Currently, work is being done

in defining the XML processing model requirements. Another issue being addressed is that of identifiers. XML does not have a document-type independent way of referencing the ID of various elements. A mechanism which allows unique element identifiers to be recognized by all conforming XML processors is crucial in making XML sub-resource linking robust (Dumbill).

Several organizations have been involved in the development of XML standards, including the W3C, ISO, and OASIS. However, there are a small number of standards developed outside of the W3C that have achieved worldwide success. There are both good and bad aspects to pursuing standards development outside of the W3C. The good aspect is the freedom to challenge new and existing XML standards. An example of this has been the development of the RELAX NG schema language which has proven to be a compelling contrast to W3C XML Schema. The bad aspect of operating outside of the W3C is a loss of coordination among web architecture and the time-tested processes of the W3C (Dumbill).

At this point in time, one of the major standards problems in XML is the need for implementing metadata storage and manipulation. Tools such as RDF and OWL, Topic Maps, and W3C XML Schema all have the ability to support this. However, the problem exists as to which terms, schemas, and ontologies to use. It is not clear for a majority of the metadata applications currently available (Dumbill).

CHAPTER V:

CASE STUDIES

NASDAQ Stock Market, Inc.

NASDAQ is the world's largest electronic stock market which lists the securities of more than 4,000 of the world's leading companies. Trades are executed through NASDAQ's sophisticated computer and telecommunications network, thus, enabling the exchange of real-time quote and trade data to more than 1.3 million users in 83 countries. NASDAQ's goal is to build the first truly global securities market that links users from all over the world. A key factor influencing the success of such a market is the timely availability of financial data pertaining to NASDAQ's member companies. Unfortunately, in today's world, investors and analysts need to compile and analyze financial data from a wide variety of sources and formats. This can be extremely difficult and time-consuming ("Microsoft Customer Evidence: NASDAQ Stock Market, Inc.").

A solution was designed and implemented which converts financial information into Extensible Business Reporting Language (XBRL) files for storage in NASDAQ's XBRL database. The database is then available to participating companies over the Internet using XML Web Services. The data is accessed by investors through the use of a Microsoft Excel workbook from which the data can be analyzed and parsed and reports can be generated ("Microsoft Customer Evidence: NASDAQ Stock Market, Inc.").

The benefits of this system are increased speed and reliability of the filing process, resulting in increases in the efficiency of the market as a whole. Also,

accessing the XBRL information directly in Excel saves steps and enables investors to work with the data in a familiar, powerful environment. Finally, because it is freely licensed and widely accepted within the industry, the use of XBRL within the NASDAQ solution permits the exchange of financial information across all software platforms and technologies ("Microsoft Customer Evidence: NASDAQ Stock Market, Inc.").

Novell, Inc.

Novell was an early and enthusiastic user of the SGML technology for its online and documentation efforts before switching over to XML in the mid-1990s. Novell has in many ways continued to push the envelope on what XML can do. For example, Novell created an XML application called DirXML to support an initiative that the company named Zero Day Start (ZDS) (Tittel, Pitts, and Boumphrey).

This initiative enables Novell's Information Systems and Technology staff to set up system and network access for new employees. It facilitates the easy management of staff ID information, security badges, telephone assignments, and a lot more. It also makes changing address and telephone information simple and straightforward for when employees move from one office to another (Tittel, Pitts, and Boumphrey).

The cornerstone of ZDS is based on Novell Directory Services (NDS), which is a network service that manages employee identity information. The XML technology extracts data from NDS and delivers it to other systems including: PeopleSoft, a recruiting system, a facilities management database, Novell's internal private branch exchange (PBX) phone system, and Novell's help desk system. Even though NDS acts as the primary repository for employee records, XML provide the technology that

supports data interchange among all the other systems involved (Tittel, Pitts, and Boumphrey).

The benefits of the ZDS system are rather impressive. The system changes telephone connections for new employees and existing employees entirely through software, thus eliminating set-up costs and reducing the need for a large technical support staff. Furthermore, because of the DirXML application, the Information Systems and Technology organization has been able to manage data flow with fewer resources, giving them the opportunity to redirect highly-trained technical resources to other key projects (Tittel, Pitts, and Boumphrey).

IBM Corporation

There are a number of uses for XML within a company as vast and diverse as IBM. One example is the use of XML within the IBM Global Procurement organization, which is the group within IBM that is responsible for managing the supply chain for the company's computer and manufacturing operations. This group works with IBM suppliers to order and manage product component pieces and parts. It also works with IBM contractors who put the pieces together according to each customer's specifications (Tittel, Pitts, and Boumphrey).

IBM operates a production procurement supply chain which covers the procurement, shipping, tracking, and payments of components used in the company's just-in-time manufacturing operations. Just-in-time operations streamline the processes that specify, design, order, ship, and manage payments for various partners involved in

the supply chain process. The just-in-time process also enables IBM to keep its stock low, only ordering components when necessary (Tittel, Pitts, and Boumphrey).

IBM's manufacturing partners range from small, simple companies to large, sophisticated companies. Therefore, IBM operates three different types of systems to support its partners' varying technical capabilities and sophistication. They include: a Web-based interface, an EDI environment, and an XML procurement management application called RosettaNet (Tittel, Pitts, and Boumphrey).

RosettaNet is an open XML standard that has been widely adopted by high-tech manufacturing organizations around the world. It creates a transaction-handling model that includes metadata about specific types of information exchange, in contrast to EDI, which provides only a mechanism for handling data exchange. RosettaNet also keeps track of the status and completion of all transactions in which data exchange is only a small part of. Many experts believe that given the complexity that RosettaNet manages, it is extremely cost-efficient and is absolutely essential for high-tech manufacturing organizations (Tittel, Pitts, and Boumphrey).

Rochester Institute of Technology PUB

Rochester Institute of Technology PUB is a student led organization for graphics and publishing oriented majors. It is dedicated to exploring the graphic communications industry using a hands-on approach. PUB members participate in seminars, tutoring sessions, guest speakers, and field trips. Each year, students also complete several projects that benefit the organization while advancing their knowledge and skills. In

2002, they produced print, Web, and wireless versions of a member directory with the help of Adobe software ("RIT PUB").

The challenges that PUB faced were to strengthen the PUB community, ensure efficient production of the directory, and learn more about network publishing. To meet these challenges, the PUB team took advantage of the network publishing capabilities and integration of four Adobe programs. The team produced the directory's layout in InDesign. They also tagged the content in XML; thus allowing PUB to export various XML files for integration into print, Web, or wireless publishing workflows ("RIT PUB").

PUB members were asked to visit a dynamic Web site, which was created using GoLive, to enter their information into an online form. This form was connected to a MySQL database. The directory team then extracted an XML file from the database and integrated the XML data into the tagged InDesign template, where the data was then checked for consistency and correctness. Three versions of the directory where then exported from the InDesign file. One of these was an XML version that flows into a GoLive template for Web publishing. Another was a low-resolution Adobe Portable Document Format (PDF) file optimized for PDAs. The third was a high-resolution Adobe PDF file suitable for printing ("RIT PUB").

There are numerous benefits of this system including the following: production is efficient and automated, members can create and update their own profiles, and PUB members gain invaluable skills from the experience. Students were able to utilize the GoLive Dynamic Content feature, allowing them to publish XML files that were exported from InDesign on the Web. In addition, InDesign automatically generated tables of contents, hyperlinks, and bookmarks for the Adobe PDF version, thus providing rich

interactivity and easy navigation. Furthermore, PUB member photos were scanned in using Photoshop and simply brought into the InDesign template as is. In general, the process of generating content and maintaining an updated directory for PUB has become almost effortless ("RIT PUB").

CHAPTER VI:

CONCLUSION

The Extensible Markup Language (XML) has developed into a widely accepted standard for representing and exchanging data on the Web and elsewhere. Developers almost everywhere are using XML to create systems that capture, manipulate, and exchange a wide variety of documents and data, ranging from electronic commerce transactions to family trees. Over the last several years, XML has become one of the most important developments in document syntax in the history of computing.

The history of markup conventions has had an enormous impact on the design and development of XML. Markup languages such as GML, SGML, and HTML have paved the way for the success of XML. The W3C, as well as other standards organizations, have actively participated in the development and ongoing maintenance of XML and its related specifications. A number of major corporations such as Microsoft, IBM, Oracle, Sun Microsystems, and various government agencies have also contributed significant resources to the current use and future development of XML and related technologies.

The enthusiastic acceptance of XML by the Internet community has played a significant role in the development of an entire family of XML standards. These standards include XML Linking Language (including XLink, XPath, and XPointer), CSS, XSL (including XSLT and XSL-FO), DOM, XML Namespaces and XML Schemas. The 'XML family' also includes a number of recently developed XML-based languages including XQuery and XInclude.

Industry consortiums play an important role in the development and adoption of XML-based vocabularies. The number of industry-specific standards such as XHTML, SVG, MathML, DocBook, and XBRL is rapidly growing. Furthermore, the growth of XML-based standards is expected to continue in the months and years to come.

XML can be used in a wide variety of roles delivering information for everything from documents to databases to metadata. The open-ended nature of XML has made it useful to implement in a wide variety of situations. There are many instances within corporate America where the implementation of XML applications have boosted productivity and controlled costs.

XML has been met with a few challenges. Some issues have been addressed and some have yet to be resolved. Nonetheless, there are many qualities that make XML a truly unique and extraordinary technology that will continue to have an impact on the computing world for a long time to come.

APPENDIX A:

XML WORKING GROUPS

This appendix provides details about the individual Working Groups within the XML Activity.

The XML Coordination Group is made up of the chairs of the individual W3C Working Groups. Its primary responsibility is to coordinate the activities of the working groups to avoid conflicts. It also suggests policy changes and maintains the W3C document that describes the XML Activity (Navarro, White, and Burman). In addition, the group provides a forum for coordination between the Working Groups of the XML Activity, between the XML Activity and other parts of the W3C, and between the XML Activity and other organizations ("Extensible Markup Language").

The XML Core Working Group is responsible for supporting the XML Recommendation, maintaining new versions such as XML 1.1 and ongoing revisions such as XML 1.0 Third Edition. Its support extends to errata and several other closely related specifications. Some of which include the XML Information Set, Namespaces in XML, and XInclude ("Extensible Markup Language").

The mission of the **XML Query Working Group** is to provide flexible query facilities to extract data from real and virtual documents on the Web. The working group also maintains a number of documents. Some of which are published jointly with the XSL Working Group ("Extensible Markup Language").

The XSL Working Group recently joined the XML Activity in 2003. They develop and maintain XSL Transformations (XSLT) for transforming XML documents, XSL Formatting Objects (XSL-FO) for formatting XML documents, and, working jointly

with the XML Query Working Group, the XML Path Language (XPath) 2.0. In addition, the group maintains a number of documents which are published jointly with the XML Query Working Group ("Extensible Markup Language").

The XML Binary Characterization Working Group, which was formed in March of 2004, is responsible for gathering information on instances where the overhead of generating, parsing, transmitting, storing, or accessing XML data may be deemed too vast for an application. It is responsible for characterizing the properties that XML provides as well as those required by the use case. It is also responsible for establishing measurements to determine whether XML 1.x and binary encodings provide the required solution ("Extensible Markup Language").

The XML Schemas Working Group is responsible for addressing means for defining the structure, content and semantics of XML documents. The group considered several submitted proposals for XML-based schema languages and published a requirements document in early 1999. The XML Schema specification became a Recommendation in May of 2001 ("Extensible Markup Language").

The XML Linking Working Group has completed its work and is no longer active. According to the W3C Draft Specification, the responsibility of this working group was to design "advanced, scalable, and maintainable hyper-linking and addressing functionality for XML". XML Linking and XML Base reached Recommendation status on June 27th, 2001. The XML Pointer Language (XPointer) reached Recommendation status on March 25th, 2003 ("XML Linking").

BIBLIOGRAPHY

1. "About the World Wide Web Consortium (W3C)." 2004. W3C. 8 Apr. 2004 <http://www.w3.org/Consortium/>.
2. "All About the Hebrew Alphabet." 2003. Torah Tots, Inc. 1 June 2004 <http://www.torahtots.com/alefbet/nekudot/allabouthebrew.htm>.
3. Bednarz, Ann. "New speech technologies making noise." Network World 22 Mar. 2004: 14. ABI/INFORM Global. ProQuest. RIT Lib., Rochester, NY. 10 May 2004 <http://wally.rit.edu/electronic/abi/abiweb.html>.
4. Bednarz, Ann. "XML security integration worries AmEx." Network World 24 Mar. 2003: 10. ABI/INFORM Global. ProQuest. RIT Lib., Rochester, NY. 13 May 2004 <http://wally.rit.edu/electronic/abi/abiweb.html>.
5. Birbeck, Mark, et al. Professional XML, 2nd Edition. New York, NY: APress, LLC, 2004. Books24x7. RIT Lib., Rochester, NY. 10 May 2004 <http://wally.rit.edu/electronic/books24x7/books24x7.html>.
6. Cascading Style Sheets, Level 2. 1998. W3C. 7 Apr. 2004 <http://www.w3.org/TR/REC-CSS2/>.
7. Chamberlin, D. "XQuery: An XML query language." IBM Systems Journal 41.4 (2002): 597-615. ABI/INFORM Global. ProQuest. RIT Lib., Rochester, NY. 10 May 2004 <http://wally.rit.edu/electronic/abi/abiweb.html>.

8. Daconta, Michael C., Leo J. Obrst, and Kevin T. Smith. The Semantic Web: A Guide to the Future of XML, Web Services, and Knowledge Management. Indianapolis, IN: Wiley Publishing, Inc., 2003. Books24x7. RIT Lib., Rochester, NY. 7 Jan. 2004 <http://wally.rit.edu/electronic/books24x7/books24x7.html>.

9. Daum, Berthold and Udo Merten. System Architecture with XML. San Francisco, CA: Morgan Kaufmann Publishers, 2003. Books24x7. RIT Lib., Rochester, NY. 26 Feb. 2004 <http://wally.rit.edu/electronic/books24x7/books24x7.html>.

10. Dumbill, Edd. "The State of XML." XML.com. 21 April 2004. O'Reilly Media, Inc. 26 April 2004 <http://www.xml.com/pub/a/2004/04/21/state.html>.

11. Extensible Markup Language (XML). 2004. W3C. 2 Apr. 2004 <http://www.w3.org/XML/>.

12. Extensible Stylesheet Language (XSL). 2001. W3C. 7 Apr. 2004 <http://www.w3.org/TR/xsl/>.

13. "Fact Sheet: W3C Issues XML 1.0 as a Recommendation." 1998. W3C. 18 Mar. 2004 <http://www.w3.org/Press/1998/XML10-REC-fact.html>.

14. Ford, Paul. "Marking Up Bureaucracy." XML.com. 24 Sept. 2003. O'Reilly Media, Inc. 19 Mar. 2004 <http://www.xml.com/pub/a/2003/09/24/government.html>.

15. Hall, Robert. "XML Support in Oracle Technology." Oracle Magazine. Sept. 2001. Oracle Corporation 4 May 2004 <http://www.oracle.com/oramag/oracle/01-sep/index.html?o51otn.html>.

16. Harold, Elliotte Rusty. Effective XML: 50 Specific Ways to Improve Your XML. Boston, MA: Addison-Wesley Pub. Co, 2003.

17. Harold, Elliotte Rusty. XML Bible, Second Edition. New York, NY: Hungry Minds, Inc., 2001. Books24x7. RIT Lib., Rochester, NY. 19 May 2004 <http://wally.rit.edu/electronic/books24x7/books24x7.html>.

18. Harold, Elliotte Rusty and W. Scott Means. XML in a Nutshell, 2nd Edition. Sebastopol, CA: O'Reilly & Associates, Inc., 2002. Safari Tech Books Online. ProQuest. RIT Lib., Rochester, NY. 9 May 2004 <http://wally.rit.edu/electronic/safari/safari.html>.

19. "HyperText Markup Language (HTML) Home Page." 2004. W3C. 30 Mar. 2004 <http://www.w3.org/MarkUp/>.

20. International, Dr. Developing International Software, 2nd Edition. Redmond, WA: Microsoft Press, 2003. Books24x7. RIT Lib., Rochester, NY. 19 May 2004 <http://wally.rit.edu/electronic/books24x7/books24x7.html>.

21. "ISO – International Organization for Standardization – Homepage." 2004. ISO. 12 Apr. 2004 <http://www.iso.ch/iso/en/ISOOnline.frontpage>.

22. Ivanov, Ivelin. "XQuery Implementation." XML.com. 1 Oct. 2003. O'Reilly Media, Inc. 15 Dec. 2003 <http://www.xml.com/lpt/a/2003/10/01/xquery.html>.

23. Kay, Michael H. "XML Five Years On: A Review of the Achievements So Far and the Challenges Ahead." Document Engineering: Proceedings of the 2003 ACM symposium on Document engineering (2003): pgs. 29-31. The ACM Digital Library. RIT Lib., Rochester, NY. 27 Feb. 2004 <http://wally.rit.edu/electronic/acm/acm.html>.

24. Kim, Larry. The Official XMLSPY Handbook. Indianapolis, IN: Wiley Publishing, Inc., 2003. Books24x7. RIT Lib., Rochester, NY. 29 May 2004 <http://wally.rit.edu/electronic/books24x7/books24x7.html>.

25. Kontzer, Tony. "Federal agencies aim to cut paperwork." InformationWeek 27 Oct. 2003: pg. 30. ABI/INFORM Global. ProQuest. RIT Lib., Rochester, NY. 3 May 2004 <http://wally.rit.edu/electronic/abi/abiweb.html>.

26. Kotok, Alan. "U.S. Federal XML Guidelines." XML.com. 6 Feb. 2002. O'Reilly Media, Inc. 21 Apr. 2004 <http://www.xml.com/pub/a/2002/02/06/fedguidelines.html>.

27. Krill, Paul. "Too many Web services standards bodies?" InfoWorld. 10 Dec. 2002. International Data Group. 12 Apr. 2004 <http://www.infoworld.com/article/02/12/10/021210hnstandards_1.html>.

28. Luening, Erich. "W3C makes XML a standard." CNET News.com. 10 Feb. 1998. CNET Networks, Inc. 27 Apr. 2004 <http://news.com.com/2100-1023_3-208004.html>.

29. Mathematical Markup Language (MathML) Version 2.0 (Second Edition). 2003. W3C. 7 May 2004 <http://www.w3.org/TR/2003/REC-MathML2-20031021/>.

30. "Media Alert: W3C Issues XML 1.0 as a Proposed Recommendation." 1997. W3C. 19 Apr. 2004 <http://www.w3.org/Press/XML-PR>.

31. "Microsoft Co-Submits Another Web Services Specification to W3C." Microsoft.com. Redmond, Wash., 15 Mar. 2001. Microsoft Corporation. 28 Apr. 2004 <http://www.microsoft.com/presspass/press/2001/mar01/03-15WebServicesPR.asp>.

32. "Microsoft Customer Evidence: NASDAQ Stock Market, Inc." Microsoft.com. 15 Jan. 2003. Microsoft Corporation. 20 May 2004. <http://www.microsoft.com/resources/casestudies/CaseStudy.asp?CaseStudyID=13543>.

33. Morejon, Mario. "Hurdles abound in the path to XML integration." CRN 20 Oct. 2003: 60. ABI/INFORM Global. ProQuest. RIT Lib., Rochester, NY. 5 May 2004 <http://wally.rit.edu/electronic/abi/abiweb.html>.

34. National Electronic Commerce Coordinating Council (NECCC). An Introduction to XML's Potential Use Within Government. Dec. 2000. 21 Apr. 2004 <http://www.ec3.org/Downloads/2000/XML_Paper.pdf>.

35. Navarro, Ann, Chuck White, and Linda Burman. Mastering XML. Alameda, CA: Sybex, 2000. Books24x7. RIT Lib., Rochester, NY. 25 Feb. 2004 <http://wally.rit.edu/electronic/books24x7/books24x7.html>

36. "OASIS – Who We Are – Mission." OASIS. 12 Apr. 2004 <http://www.oasis-open.org/who/>.

37. "Press Release: W3C Issues XML 1.0 as a Recommendation." 1998. W3C. 18 Mar. 2004 <http://www.w3.org/Press/1998/XML10-REC>.

38. Phillips, Lee Anne. Special Edition Using XML. Indianapolis, IN: Que, 2000. Books24x7. RIT Lib., Rochester, NY. 23 Feb. 2004 <http://wally.rit.edu/electronic/books24x7/books24x7.html>.

39. Rapoza, Jim. "XML: New Options, New Worries; Don't overestimate the standard's and Web services' capabilities." eWeek 23 Sep. 2002: 26H. ABI/INFORM Global. ProQuest. RIT Lib., Rochester, NY. 13 May 2004 <http://wally.rit.edu/electronic/abi/abiweb.html>.

40. Ray, Erik T. Learning XML, 2nd Edition. Sebastopol, CA: O'Reilly & Associates, Inc., 2003. Safari Tech Books Online. ProQuest. RIT Lib., Rochester, NY. 15 Mar. 2004 <http://wally.rit.edu/electronic/safari/safari.html>.

41. RDF Primer. 2004. W3C. 7 May 2004 <http://www.w3.org/TR/REC-rdf-syntax/>.

42. "Rochester Institute of Technology PUB: RIT student group showcases network publishing efficiency with Adobe GoLive and Adobe InDesign software." Adobe.com. 2003. Adobe Systems Incorporated. 4 May 2004 <http://www.adobe.com/products/golive/pdfs/rit_pub_ss.pdf>

43. Salz, Rich. "XKMS does the heavy work of PKI." Network World 8 Sep. 2003: 39. ABI/INFORM Global. ProQuest. RIT Lib., Rochester, NY. 11 Dec. 2003 <http://wally.rit.edu/electronic/abi/abiweb.html>

44. Scalable Vector Graphics (SVG) 1.1 Specification. 2003. W3C 6 May 2004 <http://www.w3.org/TR/SVG/>.

45. Tittel, Ed, Natanya Pitts, and Frank Boumphrey. XML for Dummies, 3rd Edition. New York, NY: Hungry Minds, Inc., 2002. Books24x7. RIT Lib., Rochester, NY. 23 Feb. 2004 <http://wally.rit.edu/electronic/books24x7/books24x7.html>.

46. Voice Extensible Markup Language (VoiceXML) Version 2.0. 2004. W3C. 11 May 2004 <http://www.w3.org/TR/2004/REC-voicexml20-20040316/>.

47. White, Chuck, Liam Quin, and Linda Burman. Mastering XML Premium Edition. Alameda, CA: Sybex, 2001. Books24x7. RIT Lib., Rochester, NY. 12 Apr. 2004 <http://wally.rit.edu/electronic/books24x7/books24x7.html>.

48. "Windows to Include Industry's Most Complete Implementation of XML and XSL." Microsoft.com. Denver, CO., 13 Oct. 1998. Microsoft Corporation. 28 Apr. 2004 <http://www.microsoft.com/presspass/press/1998/Oct98/XMLcapPR.asp>.

49. "W3C Synchronized Media Home page." 2004. W3C. 7 May 2004 <http://www.w3.org/AudioVideo/>.

50. W3C XML Key Management. 2004. W3C. 10 May 2004 <http://www.w3.org/2001/XKMS/>.

51. W3C XML Query (XQuery). 2004. W3C. 10 May 2004 <http://www.w3.org/XML/Query>.

52. W3C XML Pointer, XML Base and XML Linking. 2003. W3C. 1 Apr. 2004 <http://www.w3.org/XML/Linking>.

53. W3C XML Schema. 2004. W3C. 19 Mar. 2004 <http://www.w3.org/XML/Schema>.

54. "What is FpML?" 2004. ISDA - International Swaps and Derivatives Association, Inc. 7 May 2004 <http://www.fpml.org/what-is-fpml/index.asp>.

55. "What is XBRL." 2002. XBRL International. 7 May 2004 <http://www.xbrl.org/whatisxbrl/>.

56. "XML at Sun: FAQs." 2004. Sun Microsystems. 27 Apr. 2004 <http://wwws.sun.com/software/xml/faqs.html>.

57. "XML in 10 points." 2003. W3C. 19 Mar. 2004
<http://www.w3.org/XML/1999/XML-in-10-points>.

58. "XML in Mozilla." 2003. The Mozilla Organization. 19 May 2004
<http://www.mozilla.org/newlayout/xml/>.

59. XML Inclusions (XInclude) Version 1.0. 2004. W3C. 10 May 2004
<http://www.w3.org/TR/xinclude/>.

60. XML Information Set (Second Edition). 2004. W3C. 30 Apr. 2004
<http://www.w3.org/TR/xml-infoset/>.

61. XML Linking Language (XLink) Version 1.0. 2001. W3C. 7 Apr. 2004
<http://www.w3.org/TR/xlink/>.

62. XML Path Language (XPath). 1999. W3C. 7 Apr. 2004
<http://www.w3.org/TR/xpath>

63. XML Pointer Language (XPointer) Version 1.0. 2001. W3C. 7 Apr. 2004
<http://www.w3.org/TR/WD-xptr>

64. XSL Transformations (XSLT). 1999. W3C. 7 Apr. 2004
<http://www.w3.org/TR/xslt>.